Reconsidering Physical Key Secrecy: Teleduplication via Optical Decoding

Benjamin Laxton, Kai Wang and Stefan Savage
Department of Computer Science & Engineering
University of California, San Diego
La Jolla, California, USA

ABSTRACT

The access control provided by a physical lock is based on the assumption that the information content of the corresponding key is private — that duplication should require either possession of the key or *a priori* knowledge of how it was cut. However, the ever-increasing capabilities and prevalence of digital imaging technologies present a fundamental challenge to this privacy assumption. Using modest imaging equipment and standard computer vision algorithms, we demonstrate the effectiveness of physical key *teleduplication* — extracting a key's complete and precise *bitting code* at a distance via *optical decoding* and then cutting precise duplicates. We describe our prototype system, *Sneakey,* and evaluate its effectiveness, in both laboratory and real-world settings, using the most popular residential key types in the U.S.

Categories and Subject Descriptors

I.4.7 [**Image Processing and Computer Vision**]: Feature Measurement

General Terms

Experimentation, Measurement, Security

Keywords

Keys, Physical Security, Teleduplication

1. INTRODUCTION

Most of us rely on mechanical locks to physically secure our homes and places of business. We assume that these locks are challenging to open without their keys (sufficiently so that most unauthorized parties are dissuaded from trying) and that maintaining physical possession of the associated keys ensures that only the holder will make use of them. While the press and sport lock picking communities have focused considerable attention on the failings of the first assumption (e.g., the threat of key bumping) it is less widely understood that the second assumption is also increasingly being undermined by technological advances.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CCS'08, October 27–31, 2008, Alexandria, Virginia, USA.
Copyright 2008 ACM 978-1-59593-810-7/08/10 ...$5.00.

Obviously, if an adversary steals a key, even temporarily, they may then create a precise duplicate. However, duplication does not in fact require physical possession. While a key is a physical artifact, its value is entirely in the discrete pattern of cuts — the *bitting code* — that represents a secret shared with its associated lock. This information is *not* truly secret, but is exposed publicly every time a key is visible. While the difference between cut depths may be as small as 0.3mm, discerning these features is well within the capabilities of modern optics and consumer imaging technology. Indeed, it is the increasing prevalence and fidelity of such technologies — ranging from increasing deployment of video surveillance to the ubiquity of high-resolution CCD's in modern cell-phones — that motivates us to re-consider this aspect of physical key security.

In this paper, we describe a general "teleduplication" attack that allows an adversary to create precise physical key duplicates based on remote imaging data. We develop a supervised computer-vision approach for normalizing pictures of physical keys into a planar representation and then automatically extracting the associated bitting code. We have built a prototype system to do this, *Sneakey*, and we demonstrate that it is effective by decoding unknown keys, creating candidate duplicates based on those measurements and then using these duplicates to open the associated locks. Finally, we demonstrate this attack in real-world settings including cell camera captures and pictures extracted surreptitiously via telephotography.

The remainder of this paper is structured as follows: In Section 2 we review the operation of modern pin tumbler locks, existing lock bypass methods and related work related to key duplication or synthesis. We then describe our approach and the prototype system we have constructed in Section 3, followed by our preliminary results in both the lab and in real-world settings. We describe obvious improvements to our system in Section 5 and then conclude in Section 6 with a discussion of countermeasures and alternative designs to mitigate this threat.

2. BACKGROUND

The mechanical lock is perhaps the world's oldest form of access control. Locks and keys alike are mentioned throughout both Old and New Testaments, and complete artifacts have survived the ancient Egyptians, Greeks and Romans [4, 13]. The modern pin tumbler lock, derived from an Egyptian design, was invented by Linus Yale Jr. in 1861 and has changed little in its basic operation since then. A solid cylinder, the "plug", has a slot (called the keyway) cut lengthwise to accommodate a key blank of the manufacturer's design. In turn, the plug rotates freely within a solid outer shell whose inner diameter is the same as the outer diameter of the plug. The plug is mechanically attached to some entry protection mechanisms (typically a bolt via a cam) and rotating the plug is sufficient to gain entry. Thus, the lock must be designed to prevent rotation unless the proper key is inserted.

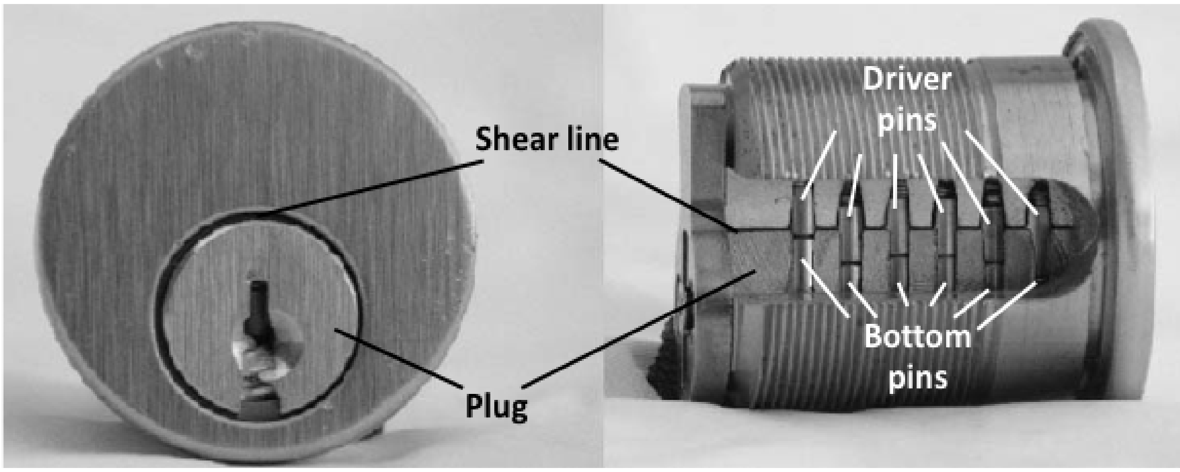


Figure 1: Front and side cutaway view of pin tumbler lock (images courtesy of Matt Blaze). When the correct key is inserted into the lock, the bottom pins are raised to the shear line and the plug is able to rotate freely.

In the pin tumbler design the plug and casing are drilled with a series of precisely spaced holes (perpendicular to the face of the lock), and each hole is then populated with a spring (backstopped by the edge of the shell) followed by two or more “pins”. The bottom pin in each hole contacts the key directly and is raised or lowered based on the depth of a key’s “cut” in that position. Thus, the bottom pins have different lengths to correspond to the different cut depths in the corresponding key. The top, or “driver”, pins are also variable length, directly in inverse to the length of the bottom pins. If a correct key is fully inserted, the division between driver and bottom pins will be placed precisely at the “shear line” between the plug and casing and the plug will be able to rotate freely; otherwise rotation will be prevented by a driver or bottom pin[1]. These structures are shown in Figure 1.

2.1 Lock manipulation

There are a wide array of techniques by which a mechanical lock may be opened without possessing a legitimate key. For example, lock picking is a well-known method whereby a shaped tool is inserted in the keyway and used to raise individual pins to the cylinder’s shear line, while tension is delivered to the cylinder to make them bind [9]. A simpler (and quicker) technique is lock "bumping" — a technique in which kinetic energy is transferred to all driver pins by hitting a specially made "bump key" cut with the deepest bitting code [14]. Both picking and especially bumping have enjoyed considerable attention in the press and lock manufacturers are ever improving mechanisms to complicate picking and bumping (e.g., via tighter machine tolerances, spool/mushroom pins, trap/rapid pins, crossbars, etc). Moreover, even if such mitigations are not foolproof they may reduce the number of attackers with sufficient skill or force even skilled attackers to spend additional time to bypass the lock, thus incurring additional risk; both picking and bumping and readily identifiable as an illicit activity if observed. Finally, neither technique is truly covert as both picking tools and bump keys leave telltale marks — either on the casing of the cylinder itself or on the pin stacks visible via micrographs [13].

[1]Master-keyed systems introduce some additional complexity in the form of an additional pin between the bottom and driver in one or more pin stacks, but the same basic principles applies.



Figure 2: A manual key decoder in use. A spring loaded probe contacts the key (here at its first cut) which turns an indicator wire (here showing a depth of 6). The goal of this paper is to achieve much the same readings, but using only digital images of keys.

2.2 Key duplication

Another approach is to create a key duplicate — either through impressioning, field casting or decoding. In the first case, a key blank is inserted into the lock and turned to bind the pins and then wiggled or tapped to “impression” pin marks on the blank edge. The blank is then filed where the marks appear and the process is repeated. Impressioning takes considerably more time than picking or bumping and also leaves tell-tale marks.

A more covert option, if a valid key can be temporarily obtained, is called key decoding, a process that relies on the discrete nature of modern keying systems. In particular, the information content in a particular key can be summarized by a set of numbers representing the depth of each cut into the blade of the key — the bitting code. Each manufacturer quantizes the depth of these cuts into a set of 8-10 cut positions where the shallowest cut (removing the least metal from a key blank) is typically 1. If a valid key can be temporarily obtained, it can be measured with physical decoding tool, which typically measures the “root depth” of each cut (the distance from the bottom of the cut to the bottom of the key blade) and associates it with the appropriate cut number (see Figure 2). This set of numbers can then be fed into a “code cutting” key duplicator along with a blank key and a precise duplicate will be replicated.

The principal technique used to complicate such key duplication is keyway restriction. Lock manufacturers patent protect the profile of their key blanks and restrict their distribution so blanks are not readily or legally available. Obviously these protections are not fundamental and in fact, keyway milling machines, notably the Easy Entrie system [6], can still produce such blanks on demand.

A more esoteric version of this duplication approach is field casting. While decoding and code cutting is a digital approach, casting is fundamentally analog. Here, a valid key is temporarily obtained and used to form a mold, which is then filled with a low-temperature alloy such as "Wood's metal"[5, 10]. This technique can obviously side-step any protection offered by restricted keyways.

However all of these approaches require physical access — either to the lock itself for impressioning or access to the correct key for decoding or casting. The motivation for this paper is to show that purely optical decoding — even at significant distance — has become a practical threat.

It is also worth mentioning two other pieces of related work. In his analysis of master-keyed locks, Blaze showed how a relatively small set of test keys could be cut to extract the bitting code for a master key [2]. This attack is particularly devastating since a master system is typically widely deployed in an organization and hence while testing can be carried out on a lock of the attacker's choosing, the master key produced will open all such locks. Worse, for many such systems, simply decoding a range of keys can reveal the master pinning due to omission. Our approach is complementary, in that it focuses on removing the need to obtain *any* physical access to keys. Thus, Blaze's master derivation attack could be mounted purely based on captured pictures of reference keys. Moreover, we are able to duplicate keys used in non-mastered systems, such as those used in typical residential locks, that are not subject to this vulnerability.

Finally, in the course of writing this paper it has become clear that we are not the first to observe that keys can be decoded optically. Indeed, some specialty locksmiths will offer to replicate a key by hand from a high-resolution photograph. For example, Keys4Classics.com advertises: "If you do not have your key code we are often able to determine it from the cuts of an existing key (email us a photo or scanned image of the key). Photos or scans should be in good focus and as close-up as possible of the key." A better known example is Ross Kinard's manual duplication, based on a public photo, of the small key used to secure the tally printer in a Diebold electronic voting machine [7]. While this particular key was very simple, we believe the basic approach generalizes to virtually all keys. In this paper we show that semi-automated decoding of popular residential keys can be performed without appreciable skill and over significant distances (over 100 feet).

3. OPTICAL DECODING

The goal of optical decoding is to transform one or more digital images of a key into its corresponding bitting code.

3.1 Assumptions

While a fully general and completely automated system would be ideal, we introduce several simplifying assumptions for our initial prototype. Among them, we assume:

1. The target key "type" is known (e.g. Schlage SC1),
2. A key face can be approximated by a 2D plane,
3. Absolute metric measurements are known for a reference image of a key,
4. A user supplies point correspondences between these reference measurements and their location in the target key.

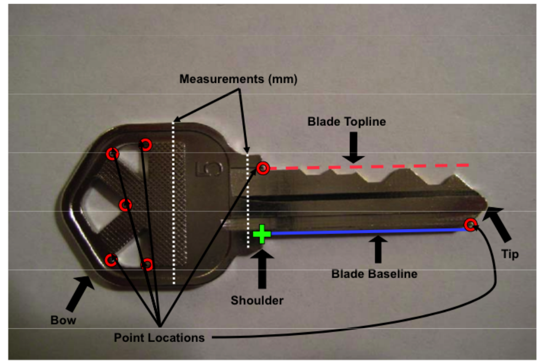


Figure 3: Above are some typical measurements for the reference key. These measurements are preformed only once for each type of key blank. The above key is a Kwikset, but we have similar reference images for other popular manufacturers.

The first of these assumptions is minor given that a small number of key types dominate given markets (e.g., for example in the U.S. residential locks are predominantly Kwikset and Schlage) and they are readily identifiable both by the shape of the key bow (although some locksmiths use vanity bows to print their contact information) and the shape of the grooves on the side of the key blade. Moreover, the lock manufacturer is typically printed on the face of the casing rendering such issues moot. Finally, for production use we believe that existing shape matching algorithms such as Belongie et al's Shape Context could be adapted to remove this assumption [1].

The second assumption also seems minor, but in fact it is not. Keys are not infinitely thin and thus any image captured from an angle not strictly perpendicular to the key profile will contain a foreshortened view of the key edge as well as the key edge. Thus, this assumption introduces error when modeling feature correspondences between keys and complicates the problem of precise edge detection (already hard for arbitrary environments), but it allows the use of simple planar transformations. Our results suggest that this is an acceptable tradeoff for a prototype.

The final two assumptions are used for normalizing a key to a plane and to an established scale. Both are easy to accomplish and require minimal skill although further automating the system remains a goal.

3.2 Key normalization

To establish a key's bitting code one must first identify where the cuts are and then establish their absolute depths. However, this is complicated for multiple reasons. First, due to key and camera position, a key may be rotated into the image plane and thus foreshortened. Second, the absolute scale of the image may not be known and hence accurate metrics may not be achievable.

We address both of these issues by normalizing the key image using multi-view geometry techniques from the computer vision field. Specifically, we make use of well known algorithms for computing a 2D homography that maps a set of points in one image to a corresponding set of points in a second image [8]. Thus, given a set of points x_i in the first image and the corresponding point locations, x'_i, in the second image, the homography captures the projective transformation that maps each x_i onto the corresponding x'_i. Under the assumption that all the points lie on a single 2D plane (Assump-

tion 2), the homography provides a way to map 2D planes onto one another. In order to compute the homography four point correspondences are sufficient, however, more correspondences can be used to refine the homography estimate and may be more robust with respect to noise in the point location labeling.

Thus, in our approach the user provides point locations on the target image that are used to compute a homography that maps the target key face to the reference key face. Since we have precise metric measurements for the reference key face we can then derive corresponding metric measurements for the target. An example of the type of measurements of a reference key we used in our system are shown in Figure 3.

3.3 Feature extraction

Given this normalization step, the only remaining tasks are to identify the two salient features: cut positions and cut depths. The first of these is simplified by the fact that two critical metric dimensions: *distance from shoulder to first cut* and *inter-cut distance* are the same for all keys using a given blank [2] and are publicly available from a range of sources [11]. For example, the first cut on a Kwikset KW-1 blank is always 0.247 inches from the shoulder and each of the subsequent cuts is spaced 0.150 inches on center. However, if this information were not available a motivated user could easily obtain it simply by manually measuring a reference key.

Identifying the bottom of the cut depth is more challenging. We have developed a heuristic to identify edges in the key profile and automatically place cut depths either on the edge itself (for flat "plateau" cuts) or at the confluence of two edges (for "valleys"). However, our initial implementation of this approach is highly error prone due to classic edge detection problems including shadowing, background noise and blur. While we continue to refine our automated algorithm, for now we allow the user to aid the system by placing depth points on a cut guideline drawn by our system (these points can be entered when they enter the initial correspondences).

Putting this all together, our step by step algorithm for decoding a key from its digital image proceeds as follows:

1. Measurements on the reference key image are taken and the pixel/mm ratio for that image is computed. This step only needs to be done once for each key blank of interest.
2. A digital image of a target key is acquired.
3. The user specifies point locations in the target key image that match those in the reference key image.
4. Using the point locations, the homography that maps the target key onto the reference key is computed.
5. Using the known *pixel/mm* ratio and the *mm* dimensions for the *distance to first cut* and *inter-cut distance*, the expected locations of each cut point along the key shaft are deterministically located.
6. A heuristic search for the depth of each key-bit can be carried out automatically or refined with user input.
7. Given the cut depth measurements for the target key in *mm* the key bitting code is given by matching the *mm* measurements to the published manufacturers specification for cut depths (e.g., a Kwikset "1" cut is 0.329 inches from the base of the key blade).

A visual depiction of the important steps in our algorithm is shown in Figure 4.

[2]This is not strictly true, as the Medeco Biaxial series can place cuts fore and aft of the standard inter-cut spacing, but this would still be easy to identify

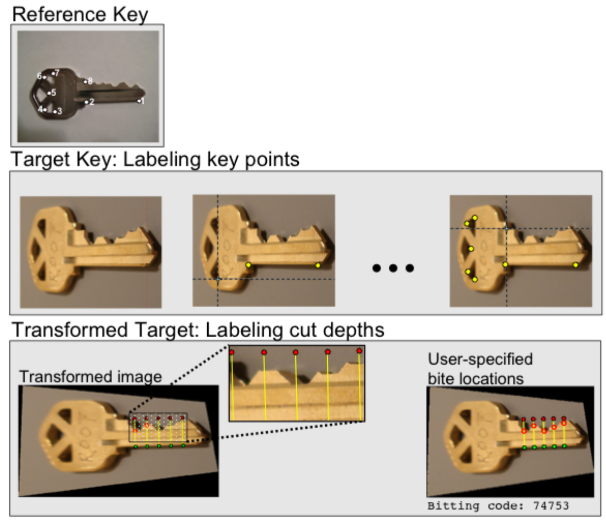


Figure 4: Above is a graphical depiction of the main steps in our algorithm for decoding a key from its image. First, the user provides point locations on the target key with a reference key as a guide. Next, the system warps the target image into the pose of the reference key and overlays markings of where the bite codes are to be found. Finally, the user specifies where the cut falls along each line and the bit depths are decoded by the system into a bitting code.

3.4 Sneakey

Our prototype system, called Sneakey, is implemented in the Matlab environment using Andrew Zisserman's VGG MultiView Geometry package [15] and presents the user with an interactive graphical interface. As previously stated, the system requires a level of user interaction, but is simplistic enough that someone with no prior knowledge about the mechanics of keys or lock mechanics can operate it.

To read the code from a key, the user is first presented with a graphical interface to select where the control points are located on the new key image. These points represent easily recognizable locations on the key, including landmarks such as corners, and are spread out across the entire key surface. Once these points have been specified, the system warps the image into the pose of the reference key. The user is presented with the original image after it has been warped into the reference pose and overlaid with marks specifying the cut locations. Along each cut location, the user specifies where along the line the bottom of the cut falls. These markings allow the system to make a measurement of the cut depth and match them with manufacturer specifications to determine bitting codes.

In principle, reading the key bitting code is simply a matter of selecting the manufacturer's depth code that most closely matches the measured depth on the new key. However, there are a range of factors that can introduce error in practice, including limited image resolution, poor optical focus, inaccurate or occluded control point placement, extreme angles, or poor lighting. Therefore, in addition to returning the most likely key code from an image, our system also returns a set of alternative key codes to help resolve ambiguities in mappings and improve the overall success rate. When this number of alternatives is small, we consider it feasible for an intruder to fashion multiple keys for breaking a single lock if it significantly increases the likelihood. [3]

[3]One of the authors consistently forgets which of his three keys is

To find alternative key codes, we consider those cut depths that are furthest away from their nearest manufacturer specified depth, and treat those codes as possibly being in one of two states: the nearest or second nearest manufacturer specified code. As a design decision, we consider the two most ambiguous cut depths in building our set of key bitting codes. With two of the five bitting codes possibly being in one of two states, we end up with an alternative set of size four, to cover all the possible code combinations. Our results in Section 4 demonstrate that the top four codes are sufficient for breaking key codes with a high success rate.

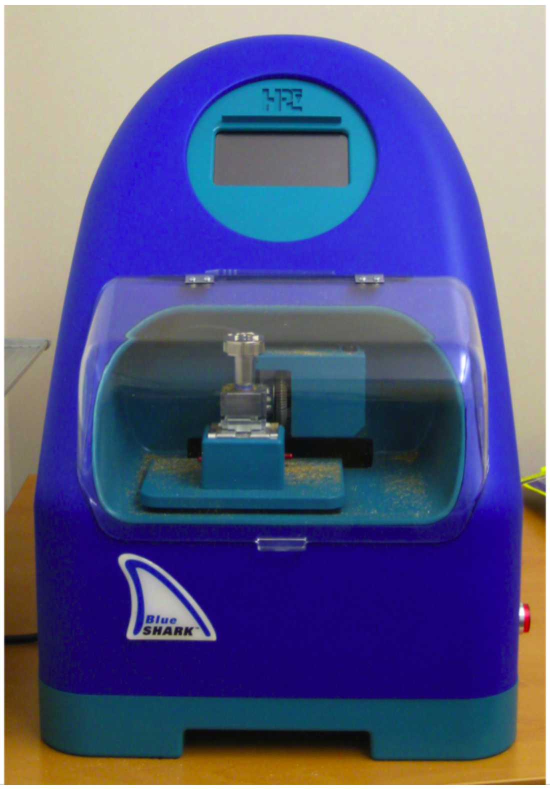


Figure 5: HPC Blue Shark code cutting machine.

4. RESULTS

We conduct three sets of experiments to evaluate our prototype. In the first, we capture key images in a controlled lab setting and investigate the impact of perspective on the accuracy of our decoding approach. In effect, this is a best case setting that provides good lighting, minimal visual noise, good lighting, and a high pixel/mm ratio. Next, we explore two scenarios for covertly capturing someone's key: a close-in scenario using a cell-phone camera and a remote capture scenario using telephotography. The former is motivated by the increasing prevalence of high-resolution cameras in modern phones, coupled with their portability and inconspicuousness.

4.1 Methodology and metrics

We consider two kinds of keys in our study: Kwikset KW-1 and Schlage SC-1. Together these keyways dominate the U.S. residential market. We believe that most pin tumbler keys will offer similar challenges. We measured the key dimensions of a reference key (for a store bought lock acquired at Home Depot) using an off-the-shelf digital caliper that claimed 0.02mm accuracy (unlikely in our experience).

For each key type, an assistant produced 15 copies cut to random depths (subject to manufacturer Maximum Adjacent Cut Specifications) using an HPC Blue Shark code cutting machine (see Figure 5) and appropriate Ilco key blanks. Of these 15, five had their code engraved on the key bow and were used for testing and refining the system on the new key blank type. The remaining 10 keys were engraved with a code number and their true codes were concealed from us until our system decoded them.

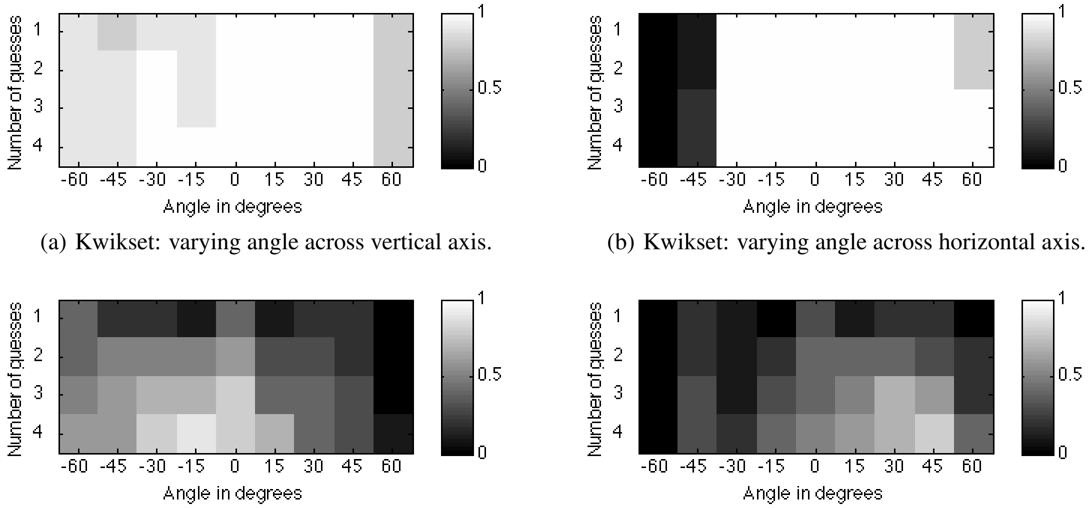
We evaluated our system using a "number of guesses" metric. Recall that our system produces a series of decodes from each key in the order that it believes are most likely to reflect the true bitting code of the key. Thus, if the first decode does not match a key but the second does, then we say that it succeeded in two guesses. In general, we consider our system to be successful if a key can be decoded within four guesses — an admittedly arbitrary threshold, but one that seems unlikely to cause alarm

4.2 Perspective tests

To understand the effect of perspective changes in the image plane, we set up an experiment where we captured each key a varying angles (using a 7.2 Megapixel Sony CyberShot DSC-W80). As its starting position, we orient the key horizontally with the teeth pointing upward and positioned on the right portion of the key, with the grip being on the left. Figure 3 shows a key in the starting position. We varied the angles at increments of 15° from 0° to 60° in both positive and negative directions across the vertical and horizontal axes. On the vertical axis, we define positive angles as those which turn the teeth away from the camera. On the horizontal axis, we define the positive angles as those which turn the teeth toward the camera. In all, this yielded 360 images. After cropping, the image sizes were typically around 1300×850 each. The images were taken at approximately one foot away from the key while it was mounted onto a surface with an adjustable viewing angle. The images were taken indoors under well lit conditions. The full results are shown in Figure 6. In each of the sub figures, the x axis represents the viewing angle and the y axis represents the number of guesses we allow to match with the true key code. Each grayscale block represents the success rate in a given setting according to its color, with 1.0 being 100%. When interpreting results from a system like this, which involves a human in the loop, one must allow for some level of noise in the results. This being the case, there are still observable patterns in the data that allow for some statements to be made.

The results exhibit a few intuitive points. First, the performance on the Kwikset keys is superior to those on the Schlage. At most angles, the performance against Kwikset are at or near 100% accurate, while performance against Schlage sees high variability and sees up to 80% accuracy in a few cases. This is likely due to the fact that the cut depths for Kwikset are spaced out further than those for Schlage, making each depth more distinct. Second, as the angle departs from the starting position of zero degrees, the performance typically degrades, with the exception of Figure 6(d). This is understandable due to the loss of information that occurs when the key is viewed at an angle. Finally, in Figure 6(b) and Figure 6(d), we see that performance degrades very sharply between the -30° and -45° angles over the horizontal axis. An analysis of the errors showed that a frequent failure was in estimating each depth as one of the true key codes. In the case of the Kwikset keys, the misread key codes were typically one above the true value (overestimating). While in the Schlage case, the missread key codes were typically

used to open which door in our building and has simply resorted to trying all three randomly.



(a) Kwikset: varying angle across vertical axis.

(b) Kwikset: varying angle across horizontal axis.

(c) Schlage: varying angle across vertical axis.

(d) Schlage: varying angle across horizontal axis.

Figure 6: Impact of perspective on key decoding effectiveness. The shade of each square represents the fraction of key images decoded correctly within the first *n* guesses (indicated on the y-axis) at a particular rotation from a straight-on horizontal viewpoint (indicated on the x-axis).

Manufacturer	1	2	3	4
Kwikset	0.8	1.0	1.0	1.0
Schlage	0.4	0.7	0.8	1.0

Table 1: Fraction of cell-phone captured key images decoded within 1, 2, 3 or 4 guesses for each key type.

one below the true value (underestimating). This implies that a central shortcoming with our method is misestimation of the key baseline or top line (as shown in Figure 3). Since the baseline and toplines are derived from the initial placement of the control points, we can see that rotations on the horizontal axis cause problems in finding the correct point locations, implying that more robust (easier to locate across changes in perspective) control points should be chosen.

The results also display some unexpected results. In both the Kwikset and Schlage tests in Figures 6(b) and 6(d), extreme positive angles in the horizontal axis had little negative effect on the outcome. In fact, in Figure 6(d) the best results were had at a +45° angle. The cause of this is likely the effect of the teeth of the keys being turned towards the camera on positive angles perhaps making the depth of the cut easier for the user to see and select.

In general, we see that performance on the Kwikset keys is very high, while performance on the Schlage keys approach high success rates when we increase the number of guesses to four.

4.3 Cell-phone tests

Our second test is motivated by the ubiquity of cell phones and the potential of increasingly high quality of cell phone cameras. The danger here is that someone can surreptitiously take a picture of a key with a modern cell phone camera and create a replica on their own time. In these experiments we used a Motorola A1200 phone which has a 2 megapixel camera. We captured one image of each test key without perspective distortions and from a distance of about 6 inches, resulting in 20 images of size 1600×1200 each. While the pixel count for the camera photos was slightly higher than that of the cropped images from the perspective tests, the image quality was considerably worse. Table 1 shows the results of the experiment. The table shows the performance against both key manufacturers with respect to the number of guesses allowed.

As with the perspective tests from the previous section, we see the performance on Kwikset is much higher than on that of Schlage, for the same reasons mentioned earlier. We also see that four guesses of key codes appear to be enough to ensure a very high level of accuracy for replication.

4.4 Telephoto decoding

So far we have focused on key decoding in the context of close-range photography — images captured within a few feet of the target. While clearly such attacks are viable, they will also require the attacker to act in close proximity to the target and potentially expose themselves to scrutiny. However, this exposure risk can be minimized through the use of telephotography. Using a sufficiently powerful optical system an attacker could, in principle, capture key images completely covertly from a significant "standoff distance".

However, there are a variety of factors that impact the range over which such long-range key decoding might be practical. We discuss these starting with theoretical limitations, pragmatic issues and then describe our own experiences.

Theoretical limitations

Roughly speaking, there are four primary qualities of interest. The effective focal length of a lens system (distance from objective to focal plane), the size of its aperture (how much light it captures), the quality of the lens itself (distortion, transmisiveness, etc) and the density of the sensor used to sample light from the focal plane.

For a given ideal lens (infinitely thin, aberration-free, perfectly transmissive, etc.) diffraction — light scattering due to self-interference — limits the size of adjacent features that can be disambiguated as a function of the focal length. This limit, called the Resolving Power, is typically modeled based on the Raleigh criterion as:

$$RP = \frac{1}{1.22\lambda N}$$

where λ is the wavelength of light and N is the f number (the ratio of the aperture to focal length) [12]. The resolving power is represented in units of line pairs per millimeter ($lpmm^{-1}$).

Another limitation is in the sensor itself. Light passing through the lens is projected onto a 2d planar array (typically a CCD or CMOS sensor) that point samples the signal based on the spacing between pixel elements (in truth, the process is more complex due to how color is processed, but our approximation is sufficient for the needs of the paper). This produces an angular resolution limit (in arcseconds per pixel) of:

$$\frac{206.265p}{F}$$

Where p is the pixel size in microns and F is the effective focal length. To provide some intuition, one arcsecond is sufficient angular resolution to provide one pixel per millimeter at a distance of 676 feet.

Thus, a particular optical system may be sensor limited or diffraction limited.

Pragmatic issues

However, there are also a wide range of pragmatic deployment limitations as well. Physically, there are practical limits to the aperture and focal length that can be easily moved and concealed. While apertures up to roughly 8 inches can be well concealed in suitcases and can be mounted on portable tripods, wider lenses are much harder to manage. Similarly, very long focal lengths present practical problems for covert deployment — its difficult to conceal an optical tube five meters long. Some of these problems can be sidestepped through the use of “folded optics” designs, which allow a doubling of focal length at the cost of some reduction in contrast, and magnification systems (again at the cost of light gain).

Another pragmatic issue is the complexity of focusing. At the resolution limit of an optical system, depth of field can become quite small and auto-focus increasingly complex to implement. Moreover, at high magnifications, the issue of camera shake becomes critical. Any touch, small air currents and even the kinetic energy of the camera’s shutter itself can cause sufficient movement to blur the image. While it is possible to build optical systems that automatically compensate (typically by moving the lenses in opposition to any motion) these become increasingly expensive to manufacture over long focal lengths. Finally, over long distances pockets of turbulent air can distort light — so-called “seeing” — and further reduce achievable resolution.

Experiences

To explore the effectiveness of over-the-counter telephotography, we obtained a Celestron C5 Schmidt-Cassegrain spotting scope (127mm aperture, 1250mm focal length) that we coupled with a TeleVue PowerMate 4X Tele-extender (an optical magnification system), producing an effective focal length of 5000mm at f/40. We mated this system with a Canon EOS 40D Digital SLR back (10.1 Megapixels at 5.7 micron pitch) mounted in prime position. This system provides an angular resolution of approximately 0.2 arcseconds per pixel resolving roughly 30 pixels per milllimeter at 100 feet. We mounted these components on a portable Manfrotto tripod with a geared head for fine grain directional adjustment. Total weight for this ensemble, shown in Figure 7 was just under 16 pounds and total cost was under $2,000.



Figure 7: Telephoto setup consisting of C5 spotting scope, Televue PowerMate 4X Tele-extender, and Cannon 40D Digital SLR. Entire system folds up into two small cases and weighs 16 pounds.

Viewing Distance	1	2	3	4
35ft	1.0	1.0	1.0	1.0
65ft	0.8	1.0	1.0	1.0
100ft	0.7	0.9	1.0	1.0

Table 2: Fraction of captured key images decoded within 1, 2, 3 or 4 guesses at each distance.

The entire apparatus was focused using a single knob on the C5 used to position the primary mirror. To address the significant issue of shake-induced blur, we took pictures using a 10 second timed delay, locked up the SLR mirror and adjusted the shutter speed and ISO for as fast an optical system as possible. A more advanced (and expensive) system could have benefited from autofocus, image stabilization and super-resolution techniques (whereby successive photos are combined to produce a higher-resolution composite).

In our telephoto experiments we captured Kwikset keys in an outdoor cafe near our department. In each experiment, we placed our target key, one of many on a ring, on a book resting on a cafe table. We mounted our camera roughly four feet above the ground oriented perpendicular to the key surface (rotations of the key out of the camera plane would strictly reduce decoding accuracy). For each of the ten keys we captured separate images at 35, 65 and 100 foot standoff distances. Figure 8 shows images of one such key at these distances, and provides a qualitative sense of how resolution degrades with distance. Manual focus also becomes more difficult, but this is not a problem at 100 feet.

Using Sneakey we decoded each of these keys with results shown in Table 2. These reults are highly robust and even at a distance of 100 feet, 7 out of 10 keys were precisely decoded within the first guess. All key shots taken at 100 feet were decoded within 3 guesses.

Finally, to explore the distance limits of our apparatus, we installed the camera on the roof of our four story department building (77 feet above the ground) at an acute angle to the key on the cafe table — 195 feet away. Adjusting focus was highly challenging because our location was unprotected from constant wind gusts that introduced motion blur. Thus we were forced to take large number of photos (roughly 40) and sift through them for the best



Figure 9: Our proof-of-concept telephoto experiment. The key image, captured at a distance of 195 feet, was correctly decoded as 74753.

ones. While time constraints on roof access prevented us from repeating the comprehensive experiments of the earlier sections, we were able to test two keys. One of these we were unable to decode (in part due to the negative impact of motion blur) while the other produced the correct code within the first three guesses (as seen in Figure 9).

5. FUTURE WORK

While we believe our early prototype has been very successful, we also found it had many limitations. In particular, we selected control points in our reference key that do not appear to be robust and we neglected to exploit inherent geometry in the way keys are cut. We plan to rework our system based on this experience and we are hopeful that it will significantly reduce error on foreshortened key images.

We would also like to extend our system to copy high-security keys such as those offered by Medeco. Medeco in particular is enticing because bitting codes are not simply depths, but also can reflect three different angles and three different positions around the cut center. Some early manual experiments suggest that, given a proper capture angle, these features should also be optically decodable.

Finally, we are interested in exploring the application of our technique to non-stationary settings. Normally motion-blur presents inhibits recognition tasks such as ours. However, since keys are rigid bodies and do not deform over time, multiple images (as from a video) can potentially be combined using "super-resolution" techniques to both deblur and enhance image quality above what any one shot could provide.

6. CONCLUSION

The security of any system invariably changes over time as technological advances challenge the system's implicit assumptions. In this paper we have identified just such an inflection point. The increasing resolution of commodity imaging sensors coupled with existing computer vision techniques has made it entirely feasible to duplicate someone's keys without ever touching them — perhaps without even being able to see them with the unaided eye. What's more, imaging has become pervasive to the point where surveillance cameras do not even produce notice. X-ray scanners, used routinely on entry to airports and government buildings, have sufficient resolution to decode keys in the same manner as well. User-generated photography only adds to this threat. Indeed, we have identified several hundred pictures of keys on the popular photo sharing site Flickr that appear to have sufficient resolution to allow duplication (we did not do so as we could not find an ethical way to validate our copies).

Given this situation, the obvious question is "what to do?". An obvious answer is "Leave your keys in your pocket". However, keys must ultimately be used — and used at *known locations*. Alternatively, keys themselves could be designed to resist optical decoding (perhaps a profile shaped like an "H", with two blades shielding each side of the bitting surface). Similarly, making keys out of transparent materials could reduce contrast and distort light in a way that would complicate decoding. However, these are all stopgap measures at best. The assumption that a code can be kept secret, in spite of being publicly exposed, is fundamentally flawed. It is probably not an acceptable long-term strategy for the technology we depend on for virtually all physical security. Inevitably keys will need to encode information which is truly secret in addition to their existing "public secrets". This is already happening. Many modern car keys include an RFID tag used to deactivate the car's engine immobilizer and Medeco's E-cylinder is extending the

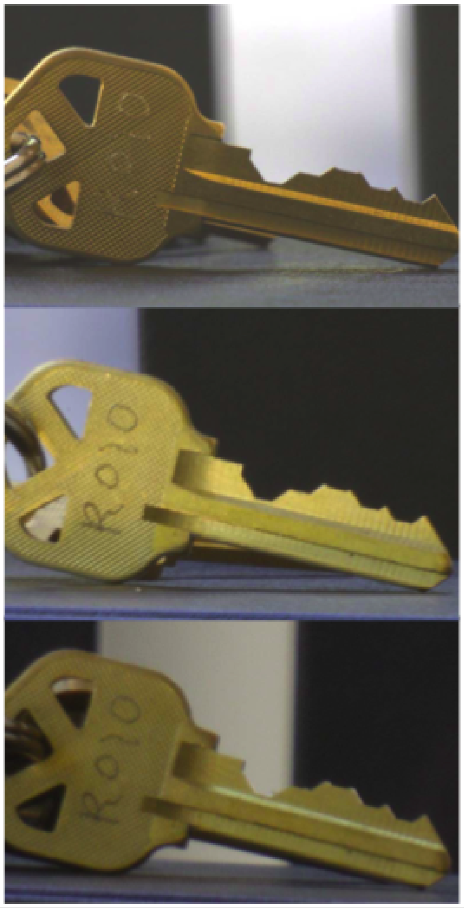


Figure 8: Sample key in telephoto experiment, captured at a distance of 35, 65 and 100 feet. The bitting code is 63134.

same approach to door locks. While these inevitably have their own security vulnerabilities [3], we suspect they will require more advanced attacks than just "looking".

7. ACKNOWLEDGMENTS

This paper was made possible through the explicit help and diligent feedback of many individuals who deserve our thanks. Brian Kantor was our resident locksmith and cut a range of keys for us on short notice. Dave Wargo provided both roof access and help with long-range photography. Serge Belongie introduced the authors and provided helpful feedback on earlier drafts of this paper. UCSD's locksmith, Clay Emison, was personally supportive of our research goals and provided key technical feedback. Kim Graves and Don Peters-Coville worked tirelessly in trying to obtain University authorization to purchase our key cutting machine while Arthur Ellis and Marianne Generales stopped the buck and ultimately provided the necessary approvals. Thanks to Craig Wetherwax and all the kind and helpful folks at Oceanside Photo and Telescope (OPT). Finally, thanks to the anonymous reviewers for their time and detailed comments.

8. REFERENCES

[1] S. Belongie, J. Malik, and J. Puzicha. Matching Shapes. In *Proceedings of the Eighth Internet Conference on Computer Vision (ICCV)*, July 2001.

[2] M. Blaze. Rights Amplification in Master-Keyed Mechanical Locks. *IEEE Security and Privacy*, 1(2):24–32, 2003.

[3] S. C. Bono, M. Green, A. Stubblefield, A. Juels, A. Rubin, and M. Sydlo. Security Analysis of a Crypographically-Enabled RFID Device. In *Proceedings of the USENIX Security Symposium*, July 2005.

[4] J. Bonomi. *Nineveh and Its Palaces: Discoveries of Botta and Layard, Applied to the Elucidation of Holy Writ (3rd ed)*. H.G Bohn, London, 1857.

[5] CIA. *CIA Field-Expedient Key Casting Manual (reprint)*. Paladin Press, 1989.

[6] Fortress Lock and Safe Co. Easy entrie. `http://www.fortresslock.co.uk/trade/index.htm`, 2008.

[7] J. A. Halderman. Diebold shows how to make your 0wn voting machine key. `http://www.freedom-to-tinker.com/?p=1113`, 2007.

[8] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[9] M. McCloud and G. de Santos. *Visual Guide to Lock Picking (3rd ed)*. Standard Publications, 2007.

[10] Multipick-Service. High tech key duplication system. `http://www.multipick-service.cc/htdocs/en/werkzeug/hightechsk/hightechsk.php`, 2008.

[11] B. Phillips. *The Complete Book of Locks and Locksmithing (6th ed)*. McGraw-Hill, 2005.

[12] S. F. Ray. *Scientic Photography and Applied Imaging*. Focal Press, 1999.

[13] M. W. Tobias. *Locks, Safes and Security: An International Police Reference (2nd Ed)*. Charles C. Thomas, Springfield, IL, USA, 2000.

[14] B. Wels and R. Gonggrijp. Bumping locks. `http://www.toool.nl/bumping.pdf`, 2005.

[15] A. Zisserman. Matlab functions for multiple view geoemtry. `http://www.robots.ox.ac.uk/~vgg/hzbook/code/`, 2008.